NRIDE: A Peer-to-Peer Ride Hailing Protocol

Martin Arrivets

martin@nride.com

v0.2.0
25 March 2023

ABSTRACT

The nRide protocol is a community-owned effort to create a more open and equitable ride-hailing ecosystem. Traditional ride-hailing applications are often plagued by high fees, a lack of transparency, and data privacy concerns. To address these issues, nRide has developed a peer-to-peer ride-hailing protocol that connects riders and drivers directly, without intermediaries. This protocol includes a messaging system, an escrow mechanism, and a driver registry, all of which are built on top of the NKN network and the JUNO blockchain. The nRide protocol is governed by a decentralized autonomous organization (DAO), and is powered by the NRIDE token, which provides access to the platform and allows holders to participate in the growth and direction of the project. This whitepaper provides an in-depth technical explanation of the nRide protocol and its key components, as well as a detailed overview of the nRide governance model and the opportunities it creates for various stakeholders.

1. Introduction

Over the past decade, the ride-hailing industry has experienced significant growth, but it has also been plagued by high fees, a lack of transparency, and data privacy concerns. Traditional ride-hailing applications operate in closed ecosystems, where their user bases are isolated from each other, leading to fragmented markets and limited opportunities for new entrants to compete with established players.

To address these issues, nRide is developing a peer-to-peer ride-hailing protocol that connects riders and drivers directly, eliminating intermediaries. By removing middlemen, nRide aims to reduce fees and increase transparency, while also creating a more equitable platform for drivers and riders. Additionally, nRide solves the problem of isolated user bases by providing a standardized way for ride-hailing applications to connect with each other and access a larger user base, fostering a more open and competitive marketplace.

nRide consists in an open-source software stack that includes a messaging protocol, an escrow mechanism, and a driver registry. The messaging protocol enables secure direct communication between riders and drivers, ensuring that all information shared between the two parties is kept private and secure. The escrow mechanism ensures that both riders and drivers fulfill their commitments by holding funds in escrow until the pickup is completed or cancelled. The driver registry is a shared database of drivers' information used by all nRide-compatible applications.

The nRide platform is powered by the NRIDE token, a dual utility and governance token that provides access to the nRide platform and its services, while also allowing holders to participate in the decentralised governance mechanism of the nRide Decentralized Autonomous Organisation (DAO). This community-driven approach ensures that the project is not controlled by any central authority or single entity, but rather governed democratically by the community. The goal of this governance model is to create a vibrant token economy of builders, users, drivers, entrepreneurs, and other stakeholders who can participate in the growth and direction of the project.

2. P2P Messaging Protocol

The messaging protocol is a key component of the nRide peer-to-peer ride-hailing platform, providing a secure and efficient means for riders and drivers to negotiate rides directly without intermediaries. The messaging protocol is built on top of the NKN network, leveraging its unique features to provide a decentralized, scalable, and robust messaging infrastructure. By defining a sequence of messages to be exchanged between riders and drivers, the messaging protocol enables efficient and reliable ride negotiation while ensuring the privacy and security of sensitive information. In this section, we will discuss the technical details of the messaging protocol, including its encryption and security features, reliability and fault tolerance, and integration with the NKN network.

2.1. Network

The implementation of peer-to-peer protocols in mobile applications can be challenging, especially due to the dynamic nature of mobile devices' network conditions, which often do not have a static public IP address. However, nRide has tackled this problem by leveraging NKN, a blockchain-based overlay network.

NKN (New Kind of Network) provides multiple independent and self-organized relay nodes, enabling clients to send and receive data for free without requiring a server or third-party services. This eliminates the need for clients to have a public IP address or port forwarding. All data transmitted through the network is end-to-end authenticated and encrypted, ensuring that only the sender and receiver can access and modify the content.

NKN's Proof of Relay (PoR) consensus mechanism incentivizes nodes to relay data packets and participate in the network by providing computing resources, bandwidth, and storage, allowing it to achieve high levels of scalability, security, and efficiency while remaining decentralized.

2.2. Messages

This section outlines the sequence of messages that are exchanged over the network to negotiate and agree on a ride in the nRide protocol. It provides a detailed description of each message and its purpose in the ride-hailing process. By following this sequence, riders and drivers can efficiently communicate with each other, agree on the details of the ride, and ensure a safe and reliable experience.

2.2.1. Request

Passengers can initiate a ride request by broadcasting a "Request" message to nearby drivers. The message contains several pieces of information, including a unique ID for the request and the coordinates of the pickup and destination points. These pieces of information allow drivers to quickly and easily identify the ride request and determine whether they are able to accept it.

Additionally, the "Request" message may include a raw "data" field that can be used to encode arbitrary information. This field may be used to include additional information about the ride request, such as the passenger's profile information or any special instructions for the driver.

```
Request {
    String id;
    String source;
    double startLat;
    double startLon;
    String startAddress;
    double endLat;
    double endLon;
    String endAddress;
    []byte data;
}
```

2.2.2. RequestCancellation

Before receiving or accepting proposals, passengers can broadcast a RequestCancellation to signify that their request is no longer active. The "acceptedProposalID" field is used to prevent multiple drivers from accepting the same ride request and to ensure that only one driver is assigned to each ride request. If a driver receives a ride request with an "acceptedProposalID" field that matches their own ID, it means that they have already accepted the request and should not take any further action. If a driver receives a ride request with an "acceptedProposalID" field that does not match their own ID, it means that another driver has already accepted the request, and the driver should disregard the request and wait for another one.

```
RequestCancellation {
    String id;
    String source;
    String requestID;
    String acceptedProposalID;
}
```

2.2.3. Proposal

When a driver receives a ride request from a passenger, they have the option to send a "Proposal" back to the sender of the request. The proposal typically includes information about the driver's position, a proposed price for the ride, and the driver's NRIDE address for setting up the escrow account. The NRIDE address is associated with a cryptocurrency wallet on the JUNO blockchain, which drivers use to send and receive NRIDE tokens when participating in the escrow mechanism for a ride request.

In some systems, this process can be automated based on the driver's preferences, location, and other factors. This means that drivers do not need to actively monitor their phones or interact with the system to analyze and respond to incoming ride requests. Instead, the system can automatically match drivers with ride requests based on various criteria, such as proximity to the passenger's location or the driver's availability.

```
Proposal {
    String id;
    String source;
    String requestID;
    double driverLat;
    double driverLon;
    double price;
    string tokenAddress;
    []byte data;
}
```

2.2.4. ProposalCancellation

Drivers can send a "ProposalCancellation" to the passenger before their proposal is accepted. This cancellation signifies that the driver's proposal is no longer valid, for example if the driver is no longer able to fulfill the ride request or if the passenger has already chosen a different driver.

Sending a ProposalCancellation allows drivers to inform the passenger that they are no longer able to fulfill the ride request, which can help to prevent confusion and ensure that the passenger is able to quickly find another driver to fulfill the request.

```
ProposalCancellation {
    String id;
    String source;
    String proposalID;
}
```

2.2.5. Acceptance

When a passenger broadcasts a ride request, they may receive multiple proposals from nearby drivers. To select the best proposal, the passenger will typically compare prices and time estimates computed locally before choosing the one that best meets their needs.

To accept a proposal, the passenger must first initialize an escrow account using the proposer's NRIDE address. This involves sending a deposit into the escrow account, which can be automated for convenience. With the escrow account initialized, the passenger can then broadcast an "Acceptance" message to nearby drivers, informing all proposers that a proposal has been accepted.

The selected proposer can then continue with the ride request, while the other drivers return to listening for requests. The escrow account ensures that the driver is paid for their services, while also providing a layer of security and protection for both the passenger and the driver.

```
Acceptance {
    String id;
    String source;
    String proposalID;
    String escrowID;
}
```

2.2.6. Confirmation

After a passenger broadcasts an "Acceptance" message to nearby drivers, the driver whose proposal was selected will receive the message and use the provided escrow ID to check the passenger's deposit in the escrow account. If the deposit is confirmed, the driver must then send their own deposit to the same escrow account before sending a "Confirmation" message to the passenger.

This process helps to ensure that both parties have a stake in the transaction and are committed to following through with the ride request. By requiring both the passenger and the driver to send deposits to the escrow account, the NRIDE ecosystem ensures that there is a mutual agreement and incentive to complete the ride.

```
Confirmation {
    String id;
    String source;
    String acceptanceID;
}
```

2.2.7. Termination

Once a "Confirmation" message has been sent by the driver and received by the passenger, either party can send a "Termination" message to signify that the ride request has been canceled or completed. This termination message may include a secret phrase that allows the other party to redeem their deposit, as well as any cancellation fees that may apply.

The use of a secret phrase helps to ensure that the refund process is secure and only authorized parties can redeem their deposit. By including this key phrase in the termination message, the other party can then use it to redeem their deposit from the escrow account.

```
Termination {
    String id;
    String source;
    String confirmationID;
    String reason; // Completed or Cancelled
    String secret;
}
```

2.2.8. TerminationAck

When a ride request is terminated by either the passenger or the driver, the other party can respond with a "TerminationAck" message that includes a secret phrase. This phrase allows the counterparty to unlock their part of the escrow account and receive their deposit, as well as any applicable cancellation fees.

By responding with a TerminationAck message that includes a secret phrase, the user helps to ensure that the refund process is secure and that the other party can redeem their deposit from the escrow account.

Overall, this process helps to create a more transparent and reliable ride sharing experience for both passengers and drivers, by providing a mechanism for canceling ride requests and receiving refunds if necessary. The use of the secret phrase also helps to ensure that the refund process is secure and that only authorized parties can redeem their deposit from the escrow account.

```
TerminationAck {
    String id;
    String source;
    String revocationID;
    String secret;
}
```

3. Escrow

The escrow mechanism is a crucial component of the nRide protocol that ensures trust and security for both riders and drivers. By locking funds in escrow until the pickup is completed, the mechanism mitigates the risk of a late cancellation or no-show. This creates a level playing field for riders and drivers and incentivizes them to fulfill their respective obligations. In the following section, we will describe the workings of the escrow mechanism in detail, including the different stages of the process and the various outputs that can result from its operation.

3.1. Functioning

The escrow smart-contract allows multiple users to create independent escrows. Each escrow involves two users and has a unique id (for future calls to reference it) as well as other parameters that guard the state transitions that are involved in the lifecycle of the escrow.

Each escrow is composed of two accounts, each guarded by what we call a secret. Once an escrow is funded by both parties, a user must obtain the other user's secret to unlock all of their deposit, otherwise they will lose a portion of it. In effect, the secret is simply the private component of a cryptographic key pair created for the purpose of the escrow. When a user funds their account in the escrow, they create a new key pair locally, and set the new public key as the 'lock' of their account in the escrow. The other user will need to obtain the private key - the secret - to unlock their deposit. Users exchange secrets offline, and the procedure for doing so is not covered by the protocol (could be by tapping phones through NFC, or by sending a message over the NKN network). The escrow smart-contract implements a state machine that guards all possible transitions and outputs given initial configuration and the messages submitted by the users.

How it works:

Both parties, A and B, lock the same amount of tokens in the smart contract. Once the funds are locked, each party can send one of two messages to the smartÂ contract:

Approve(secret)
: A message containing the counterparty's secret.

Cancel
: A message signifying the user's desire to redeem a proportion of their deposit prior to completion, or because they failed to obtain the other user's secret.

Additionally, the escrow smart-contract is configured with two timeouts:

T1
: After the first deposit is paid into the escrow account, party B has T1 time to send their deposit, after which party A will get their deposit back. T1 is left to the discretion of the first party to create and instantiate the contract. However, T1 is intended to be relatively short, only a few minutes after the agreement is initially formed and the first party engages with the contract (i.e. sending tokens to the contract).

T2 T2 is a longer timeout that triggers a default action in case either of the parties fails to submit a message to the smart contract after both deposits have been paid. T2 is intended to cover the time it would take to complete the pickup and is left to the discretion of the first party that creates the escrow.

3.2. Possible Outcomes

The following table contains the 16 possible end states of an escrow:

	Approve(B,s_a)	Cancel(B)	T1(B)	T2(B)
Approve(A,s_b)	(1, 1)	(1.3, 0.7)	x	(1.5, 0.5)
Cancel(A)	(0.7, 1.3)	(0.5, 0.5)	(1, 0)	(0.5, 0)
T1(A)	x	(0, 1)	x	(0, 1)
T2(A)	(0.5, 1.5)	(0, 0.5)	(1, 0)	(0, 0)

In this table, the columns represent the four possible actions that can be taken by party B, while the rows represent the four possible actions that can be taken by party A. Each cell in the table represents the outcome of the escrow mechanism for the combination of actions taken by both parties.

The table lists the coefficients that determine how much of the escrow funds each party receives in each possible outcome. For example, if both parties approve the transaction and exchange secrets, they both receive the same amount of funds they originally deposited (1, 1). If one party cancels and the other approves using the cancelling party's secret, the approving party receives 1.3 times their original deposit, while the cancelling party receives 0.7 times their original deposit (1.3, 0.7). If both parties cancel, they both receive 0.5 times their original deposit, and the remaining funds go to the protocol treasury (0.5, 0.5). Finally, if both parties fail to take any action after both deposits have been paid, both parties lose their deposits, and the funds go to the treasury (0, 0).

- Approve(B, secret_a) & Approve(A, secret_b))
  Both parties send each other their secrets before T2, resulting in both parties receiving their full deposit back.
- Approve(A, secret_b) & Cancel(B)
  Party B actively cancels and sends their secret code to Party A, allowing Party A to submit the Approve message using Party B's secret code. Party A receives 1.3 times their initial deposit, and Party B receives 0.7 times their initial deposit.
- Approve(A, secret_b) & T1(B)
  Not possible because the smart contract does not allow Party A to approve unless Party B's account is funded.
- Approve(A, secret_b) & T2(B)
  Party B sends their secret code to Party A, but for some reason, Party B did not cancel or send an Approve message before T2. In this case, Party B is penalized a bit more than if they had actively canceled because they made Party A wait for the expiry of the timeout. Party A receives 1.5 times their initial deposit, and Party B receives 0.5 times their initial deposit.
- Cancel(A) & Approve(B, secret_a)
  Symmetrical to output 2
- Cancel(A) and Cancel(B)
  Both parties actively cancel, but neither is able to submit the other's secret code. Both parties get penalized the same amount, and funds go to the treasury. Both parties receive

0.5 times their initial deposit.

- Cancel(A) AND T1(B)
  Party B never funded their account. After T1, Party A can withdraw their full deposit.
- Cancel(A) AND T2(B)
  Party A actively cancels, they never received Party B's secret code, and Party B timed out. This happens when one party is a no-show, and they get penalized completely.
- T1(A) AND Approve(B, secret_a)
  Not possible. Symmetrical to output 3.
- T1(A) AND Cancel(B)
  Symmetrical to output 7.
- T1(A) AND T1(B)
  Not possible. Creating an escrow requires funding the creator's account, so at least one account is funded.
- T1(A) AND T2(B)
  Party A never funded their account. Party B can Cancel before T2 to get their funds back, but even if they wait until after T2, they will still get their deposit back.
- T2(A) & Approve(B, secret_a)
  Symmetrical to output 4.
- T2(A) & Cancel(B)
  Symmetrical to output 8.
- T2(A) AND T1(B)
  Symmetrical to output 12.
- T2(A) AND T2(B)
  Both parties are penalized completely, and funds go to the treasury. Both parties receive 0 times their initial deposit.

3.3. Implications

How do users receive compensation when their counterparties cancel or fail to fulfill an obligation?

If the counterparty decides to cancel a confirmed commitment, they have two options: actively cancel and provide their secret code to the other user or do nothing. If they choose to provide their secret code, the other user can submit the Approve(secret) message to receive a cancellation fee of 1.3 or 1.5 times their initial deposit, depending on the timing of the cancellation.

The escrow mechanism is designed to encourage the counterparty to disclose their secret code as this allows them to retain a portion of their initial deposit. If they successfully submit the cancel message before T2, they will be able to retain 0.7 of their initial deposit. If they submit the cancel message after T2, they will only be able to retain 0.5 of their initial deposit. By providing an incentive for the counterparty to disclose their secret code, the escrow mechanism helps to ensure that both parties are able to recover at least a portion of their initial deposit in case of a cancellation.

How does the escrow system protect against bots programmed to automatically accept or request pickups without fulfilling commitments?

If a bot accepts or requests pickups automatically without fulfilling its commitments, it will likely be unable to recover its initial deposit from the escrow account. This is because the counterparty will not release the secret code needed to send an "Approve" message, and the bot will forfeit its deposit. Therefore, the escrow mechanism discourages bots from engaging in this type of behavior, as it imposes a financial penalty

on parties that do not fulfill their commitments.

What happens when both parties turn up at the pickup point, and one of the users receives the other's secret, but doesn't share their own secret (maliciously or for technical reasons)?

This corresponds to "Approve(A, s_b) AND Cancel(B)", or the symmetrical state, and would result in an unfair loss for the person who shared their secret. The exchange of secrets happens at pickup and in person, in the physical realm, so at that stage, refusing to share one's secret is akin to running away without paying after a regular Taxi ride. nRide users will just have to trust each other not to do that. In the eventuality of a technical problem, the affected party can always ask for offline compensation, or refuse to proceed with the journey.

4. Driver Registry

The driver registry is a fundamental component of the nRide protocol that enables riders to discover available drivers in their area. The registry is implemented as a decentralized database using blockchain technology, allowing for secure and transparent storage of driver information. Each driver entry includes their NKN address and an arbitrary location field that can be used to represent their location information in any format, giving maximum flexibility to nRide applications and users. In addition to the location field, driver entries also contain signed certificates indicating their affiliation with specific ride-hailing applications or entities. Drivers can update their location and certificate information on the registry as they move around or change affiliations, allowing riders to easily find nearby drivers and make informed choices about who to ride with.

Regarding the location field, nRide is working on a Hexagonal Hierarchical Spatial Index, which is one proposed standardized method for representing location information that provides a high level of precision while maintaining data privacy. However, since the location field is an arbitrary string, users and developers are free to define and implement their own standardized methods for organizing drivers in their area. This allows for maximum flexibility in the location field of the registry, giving developers creating applications on top of the nRide platform the ability to implement a custom location scheme to organize their drivers differently. Different location schemes correspond to different, potentially overlapping user-bases.

To ensure that riders have access to trustworthy drivers, the registry includes a certificate system that allows drivers to demonstrate their affiliation with specific ride-hailing applications or entities. Apps can filter drivers by certificate to ensure that only drivers affiliated with their app or a trusted entity are presented to their users. For example, in regions where only licensed private hire vehicles or traditional taxis are allowed to take rides, the rider can upload a signed certificate from a licensed private hire operator, and apps might broadcast requests only to drivers that have a certificate from a trusted private hire operator.

The driver registry is shared by all nRide-compatible applications, providing a common pool of drivers for riders to choose from. This allows for a more open and competitive marketplace, as riders have access to a larger pool of drivers, and apps can choose which drivers to present to their users based on specific criteria.

In summary, the driver registry is a critical component of the nRide protocol, providing a decentralized database of driver information that allows riders to easily discover available drivers in their area. By utilizing a standardized location index and certificate system, the registry ensures that riders have access to trustworthy drivers, while also allowing for maximum flexibility and customization for developers creating applications on top of the nRide platform.

5. Token

NRIDE is the native utility and governance token of the nRide platform. It serves as the underlying

currency of the escrow mechanism that protects users from late cancellations or no-shows. To use the platform as a rider or a driver, it is necessary to have NRIDE tokens.

The value of NRIDE tokens is anchored to the convenience fee for cancellations, which reflects the inconvenience caused by a late cancellation or no-show. The escrow deposit adapts to the price of the token, targeting a deposit value of $10 (indicative). As a result, the price of the token is not capped by the market value of a cancellation fee, and it can increase freely with demand.

Token holders have the opportunity to invest in the growth of the nRide ecosystem and own a stake in its financial value. They can benefit from the network's growth and adoption, as well as indirectly benefit from rides booked on any nRide-compatible application. In addition, NRIDE token holders have governance rights, which allow them to participate in the decentralized governance mechanism of the nRide DAO. The DAO controls a significant portion of the total token supply, and token holders can vote on governance proposals based on their relative holdings. The DAO can fund initiatives that benefit the community or introduce changes to the protocol, ensuring that the nRide ecosystem evolves to meet the needs of its users.

The nRide ecosystem is designed to create a vibrant token economy that encourages community participation and incentivizes growth. As the platform continues to expand and attract new users and developers, the value of NRIDE tokens is expected to increase, providing opportunities for token holders to benefit from their investment in the ecosystem. Through the combination of utility and governance features, NRIDE tokens play a key role in the success of the nRide platform, and help to ensure its long-term sustainability and growth.

6. Governance

Governance is an important aspect of the nRide ecosystem, and the nRide token is the key to decentralized decision-making. The nRide protocol is open-source, and the nRide DAO is responsible for managing the treasury and making decisions regarding the future direction of the platform. The DAO is governed by token holders, who can vote on proposals that will shape the evolution of nRide.

Token holders have the opportunity to propose changes to the protocol or initiatives that could benefit the community. Proposals can include changes to the messaging protocol, the escrow mechanism, the registry, or any other aspect of the platform. Token holders can also vote on proposals put forward by other members of the community. The voting system is based on a one-token, one-vote principle, meaning that the weight of each token is equal to the vote it represents.

The nRide DAO is designed to be as transparent and decentralized as possible, with all decisions made by community members through a consensus-driven process. The DAO's budget is financed by the initial treasury, as well as the proceeds from the escrow contract, in those cases where both users cancel or let the escrow timeout.

By putting the power of decision-making in the hands of token holders, nRide aims to create a platform that is truly owned and operated by its users. The decentralized nature of the DAO ensures that no single entity or group can dominate the decision-making process, and that the future of nRide is shaped by the community as a whole.

7. Roadmap

The nRide project was founded in October 2021 at the Gitcoin GR11 Hackathon sponsored by NKN, where it won a prize for its innovative approach to peer-to-peer ride-hailing. At the beginning, the idea was to build an uber-like service where drivers and riders connect directly to one another, without relying on a central third party. After the hackathon, NKN accepted to fund a more advanced prototype, and from there the

scope expanded to building a more generic protocol and a suite of open-source software packages that would allow the creation of compatible ride-hailing apps sharing a common pool of users.

The first whitepaper outlining this vision was written in early 2022, and the code has been under development since then. Currently, the nRide mobile application, implemented in Flutter, is in private beta on the Apple and Android app stores. The escrow and registry smart-contracts are implemented in CosmWasm and deployed on JUNO, owned and controlled by the DAO. The DAO itself is built on DAO DAO, also on JUNO, and the NRIDE token is a CW20 token, also minted and controlled by the DAO.

The short-term goals for nRide include expanding the user base of the platform, testing and refining the escrow and registry mechanisms, and improving the user experience of the mobile application. The medium-term goals for nRide include, refactoring and open-sourcing the codebase, launching a public beta version of the platform, and establishing partnerships with ride-hailing companies and other organizations.

The long-term goals for nRide include becoming the leading peer-to-peer ride-hailing platform in the world, achieving widespread adoption, and fundamentally changing the way people think about and use ride-hailing services. Key milestones for nRide include launching the public beta version of the platform, achieving a certain number of active users, establishing partnerships with major ride-hailing companies, expanding into new markets and regions, and achieving financial sustainability through revenue generation and token economics.

The nRide project is committed to achieving its goals and milestones through a combination of innovative technology, strategic partnerships, and community involvement. The DAO, built on DAO DAO on JUNO, controls the NRIDE token, a CW20 token minted and controlled by the DAO, which serves as the underlying currency of the escrow mechanism. The roadmap provides a clear vision for the future of the platform and ensures that all stakeholders are aligned and working towards a common goal.

8. Conclusion

In conclusion, nRide is a decentralized peer-to-peer ride-hailing platform that leverages blockchain technology and smart contracts to enable a trustless and secure ride-hailing experience. The platform aims to disrupt the traditional ride-hailing industry by removing the need for centralized intermediaries and empowering drivers and riders to interact directly, without the need for a third party to facilitate the transaction. The escrow mechanism, powered by the NRIDE token, ensures that both parties are held accountable for their commitments and provides compensation for any negative impact on the other party. The nRide protocol is open-source, allowing anyone to build on top of it and contribute to the growth of the ecosystem. With a strong vision for the future and a dedicated team, nRide is poised to revolutionize the ride-hailing industry and create a more equitable and efficient transportation network for everyone.

In conclusion, nRide is a pioneering decentralized ride-hailing platform that is poised to disrupt the traditional ride-hailing industry. Built on blockchain technology and smart contracts, nRide enables a trustless and secure ride-hailing experience that empowers drivers and riders to interact directly, without the need for a third party to facilitate the transaction.

The escrow mechanism, powered by the NRIDE token, is a key component of the platform, ensuring that both parties are held accountable for their commitments and providing compensation for any negative impact on the other party. The driver registry is another important component of the nRide protocol, providing a shared database of drivers' information that can be used by all nRide-compatible applications to discover available drivers in their vicinity.

The open-source nRide protocol allows anyone to build on top of it and contribute to the growth of the ecosystem, fostering innovation and collaboration. With a strong vision for the future and a dedicated team, nRide is poised to revolutionize the ride-hailing industry by creating a more equitable and efficient

transportation network for everyone. By removing the need for centralized intermediaries, nRide enables drivers and riders to keep more of the value generated by their transactions, reducing fees and increasing transparency.

The potential impact of nRide extends beyond ride-hailing, as the platform provides a model for how blockchain technology and decentralized protocols can be used to build more equitable and efficient systems across a wide range of industries. We are excited to be part of this revolution and look forward to working with our community to achieve our shared vision.